CENTRO UNIVERSITÁRIO DE ANÁPOLIS

UNIEVANGÉLICA

CURSOS SUPERIORES DE COMPUTAÇÃO

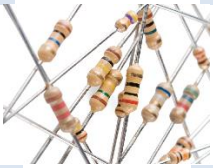
Professor: Alexandre Moraes Tannus - 2018

Arduino: Display LCD

1. OBJETIVOS:

- Conhecer os fundamentos do uso de display LCDS
- Implementar projetos com display LCD no Arduino.

2. MATERIAIS:

	Uma placa Arduino Uno
	Potenciômetro
	Resistores
	Display LCD

3. PARTE TEÓRICA

3.1. Display de Cristal Líquido (*Liquid Crystal Display* – LCD)

A utilização de *displays* é muito comum no dia a dia. Com diversos tamanhos disponíveis (Figura 1) estes visores são úteis para mostrar variados tipos de informação. Existem dois tipos básicos de visores: serial e paralelo. Estes modelos são utilizados para a visualização de caracteres. Alguns modelos mais recentes permitem mostrar imagens e gráficos (Figura 2). Nesta aula utilizaremos apenas displays de caracteres paralelos.

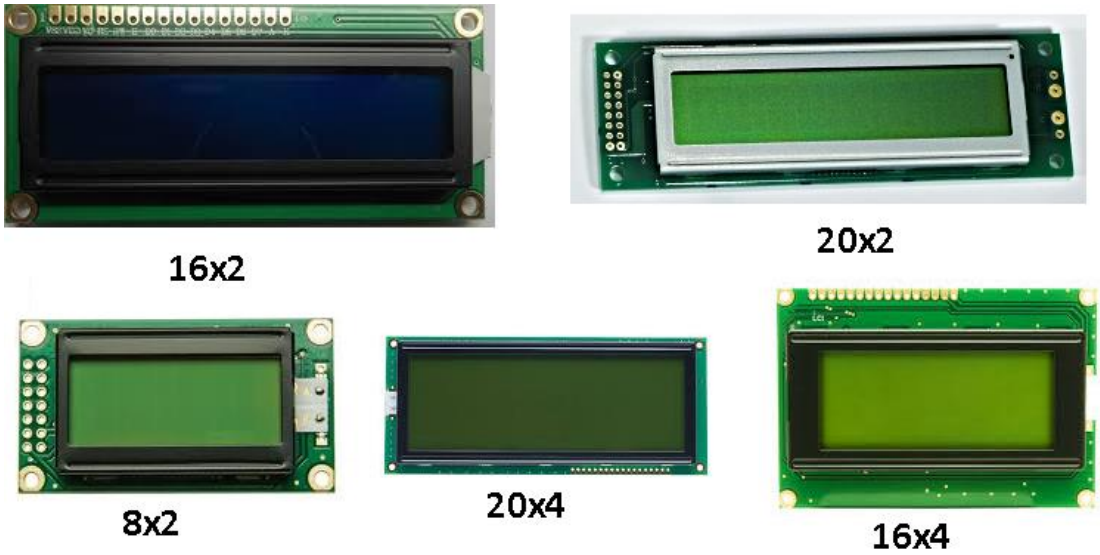


Figura 1 - Displays LCD de caracteres

Figura 2 - Display gráfico

3.2. Controlador Hitachi HD44780

Para utilizar os *displays* de cristal líquido é necessário a utilização de um controlador específico. Um dos controladores mais comuns no mercado é o Hitachi HD44780, presente em diversos modelos de LCD de vários tamanhos diferentes. Utilizaremos o modelo com 16 colunas e 2 linhas, conhecido como LCD 16x2, mostrado na Figura 3. Cada elemento (célula) pode ser acessado e alterado via programação, utilizando funções que serão explicadas na Seção 3.4. Cada caractere do *display* é formado por 40 pixels dispostos em uma matriz de 8 linhas por 5 colunas (8x5).

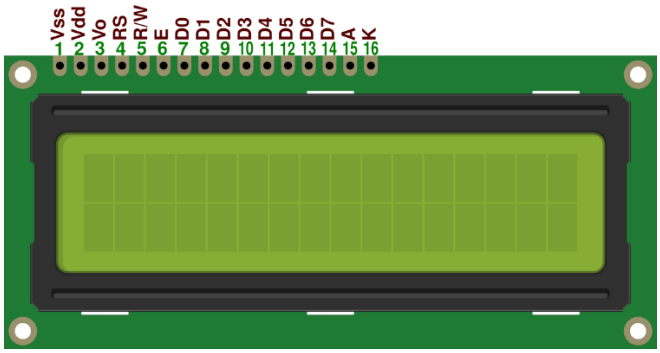


Figura 3 - Display 16x2

Alguns modelos de LCD possuem luz de fundo (*backlight*). Esta luz pode ser ativada para melhorar a visualização dos caracteres do display. Alguns modelos possuem um RGB na *backlight* e permitem a alteração da cor desta iluminação, tornando possível a criação de programas que informam uma determinada situação através da cor da luz de fundo, além do texto padrão. Os pinos 15 e 16 são utilizados para alimentar a *backlight*. É importante ressaltar a necessidade do uso de um resistor para limitar a corrente do LED. O valor desse resistor dependerá do modelo de *display* utilizado (EVANS; NOBLE; HOCHENBAUM, 2013). A alimentação da luz de fundo deve ser feita conforme mostra a Figura 4.

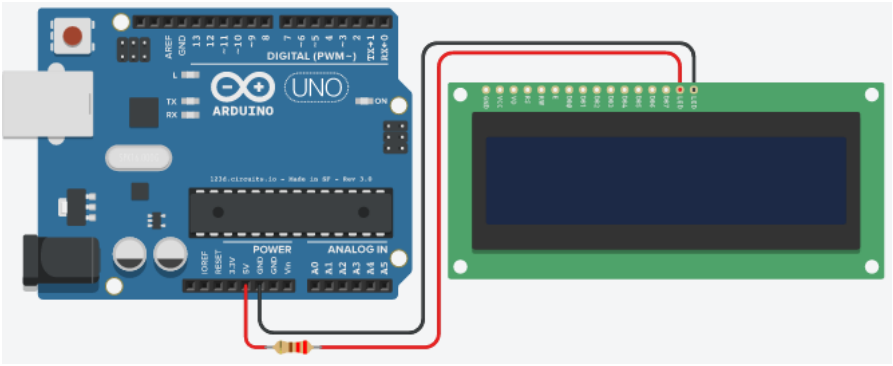


Figura 4 - Alimentação da luz de fundo

Outro ponto importante a se destacar no controlador é a possibilidade de trabalhar com 4 ou 8 *bits* de dados por vez. A diferença entre os dois modos é a quantidade de pinos utilizados para a comunicação e a velocidade de execução. Para o modo de 4 bits apenas os pinos *D4* a *D7* são utilizados. O envio do *byte* é feito em duas partes, sendo a primeira contendo os 4 *bits* mais significativos e em seguida os quatro *bits* menos significativos. No modo 8 bits todas os pinos de dados (*D0* a *D7*) são utilizados (HITACHI, 1998).

3.3. Conexão do *display* ao Arduino

Antes de conectar o LCD à placa do Arduino é importante saber o que cada pino representa para o *display*. A Tabela 1 mostra a função e o nome de todos os pinos do *display*.

Tabela 1- Pinos LCD

Pino	Nome	Função
1	**Vss**	Terra
2	**Vdd**	Alimentação positiva (5V)
3	**Vo**	Contraste do LCD
4	**RS**	*Register Select* – Define se o dado enviado ao LCD é um comando (nível lógico 0) ou um caractere (nível lógico 1)
5	**R/W**	*Read/Write* – Define se será realizada leitura (nível lógico 1) ou escrita (nível lógico 0) de dados.
6	**E**	*Enable* – Habilita recepção de sinal do LCD
7	**D0**	Dados – não utilizado em modo 4 bits
8	**D1**	Dados – não utilizado em modo 4 bits
9	**D2**	Dados – não utilizado em modo 4 bits
10	**D3**	Dados – não utilizado em modo 4 bits
11	**D4**	Dados
12	**D5**	Dados
13	**D6**	Dados
14	**D7**	Dados
15	**A (LED+)**	Anodo da *backlight*. Deve ser ligado à alimentação positiva com um resistor em série
16	**K (LED-)**	Catodo da *backlight*. Deve ser ligado ao terra.

A conexão LCD-Arduino pode ser feita conforme mostra a Tabela 2 e a Figura 5.

Tabela 2 - Conexão LCD-Arduino

Pino LCD	Pino Arduino
Vss	GND
Vdd	5V
Vo	Pino 2 do potenciômetro
RS	Pino 6
R/W	GND (opcional – *D8*)
E	Pino 7
D0	Desconectado
D1	Desconectado
D2	Desconectado
D3	Desconectado
D4	Pino 5
D5	Pino 4
D6	Pino 3
D7	Pino 2
A (LED+)	5V – com resistor 220Ω em série
K (LED-)	GND

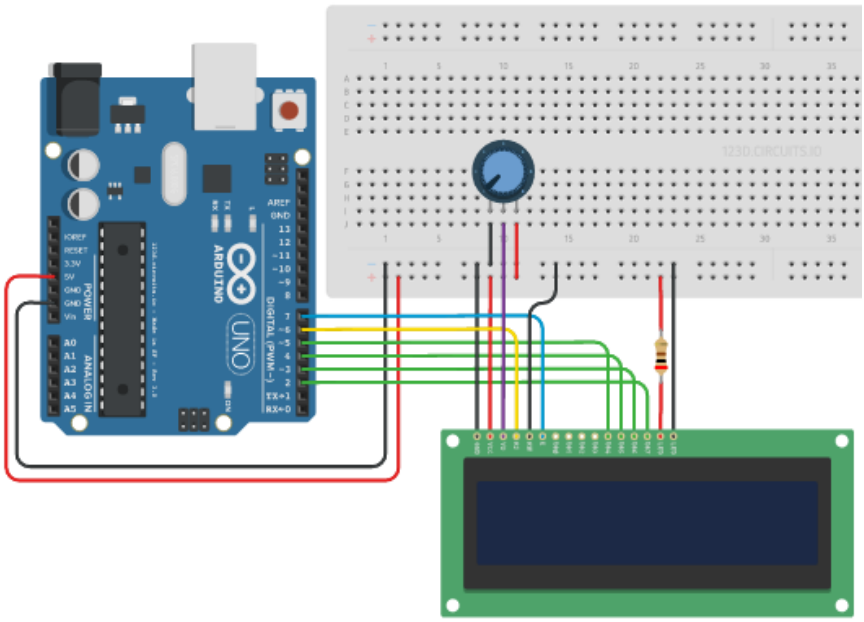


Figura 5 - Conexão LCD-Arduino

3.4. Bibliotecas e funções

Para realizar a programação do *display* 16x2 será utilizada a biblioteca *LiquidCrystal* que já está pré-instalada na IDE do Arduino. Para incluir a biblioteca no programa deve ser utilizada a diretiva

$$\#include < LiquidCrystal.h >$$

na primeira linha do código fonte. Em seguida é necessário criar um objeto através do comando

$$LiquidCrystal\ lcd(pinoRS, pinoEnable, pinoD4, pinoD5, pinoD6, pinoD7)$$

Este objeto será utilizado sempre que uma função da biblioteca for chamada no programa. As linhas iniciais do programa ficam conforme o mostrado no Código 1.

```
#define pinoRS 6
#define pinoEnable 7
#define pinoD4 5
#define pinoD5 4
#define pinoD6 3
#define pinoD7 2

#Include <LiquidCrystal.h>

LiquidCrystal lcd(pinoRS, pinoEnable, pinoD4, pinoD5, pinoD6, pinoD7)

void setup( ) {

}

void loop( ) {

}
```

Código 1 - Inclusão da biblioteca LCD

Dentre as funções presentes nessa biblioteca podem ser destacadas as listadas na Tabela 3.

Tabela 3 - Funções da biblioteca LiquidCrystal

Função	Descrição
$\boldsymbol{begin(int\ coluna, int\ linha)}$	Define a dimensão da tela
$\boldsymbol{clear()}$	Limpa a tela
$\boldsymbol{setCursor(int\ coluna, int\ linha)}$	Define a posição do cursor para a próxima excrita
$\boldsymbol{print(data)}$	Imprime um dado no *display*. O dado pode ser texto, número ou caractere criado.
$\boldsymbol{home()}$	Coloca o cursor no canto superior esquerdo.
$\boldsymbol{cursor()}$	Exibe um caractere sublinhado na posição atual
$\boldsymbol{noCursor()}$	Esconde o cursor
$\boldsymbol{blink()}$	Pisca o cursor
$\boldsymbol{noBlink()}$	Desabilita o piscar do cursor
$\boldsymbol{scrollDisplayLeft()}$	Rola o texto um espaço para a esquerda
$\boldsymbol{scrollDisplayRight()}$	Rola o texto um espaço para a direita
$\boldsymbol{autoScroll()}$	Rolagem automática do texto da esquerda para a direita
$\boldsymbol{noAutoScroll()}$	Desabilita a rolagem automática
$\boldsymbol{createChar(num, data)}$	Cria um novo caractere para exibição *num* – 0 a 7 *data* – vetor de byte com o caractere criado

Esta biblioteca trabalha de forma transparente com a transmissão em 4 ou 8 bits, ou seja, não é necessário realizar nenhuma adaptação em caso de mudança de modo de transmissão de dados.

4. PARTE PRÁTICA

4.1. Prática 01 – Escrever texto no display

Nesta prática será utilizado o *display* para mostrar um texto estático. Para realizar este experimento é necessário montar o circuito da Figura 6. Nesta montagem o potenciômetro é utilizado para controlar a luminosidade da *backlight*. O texto é escrito utilizando a função $print()$. Escreva o código presente no Código 2

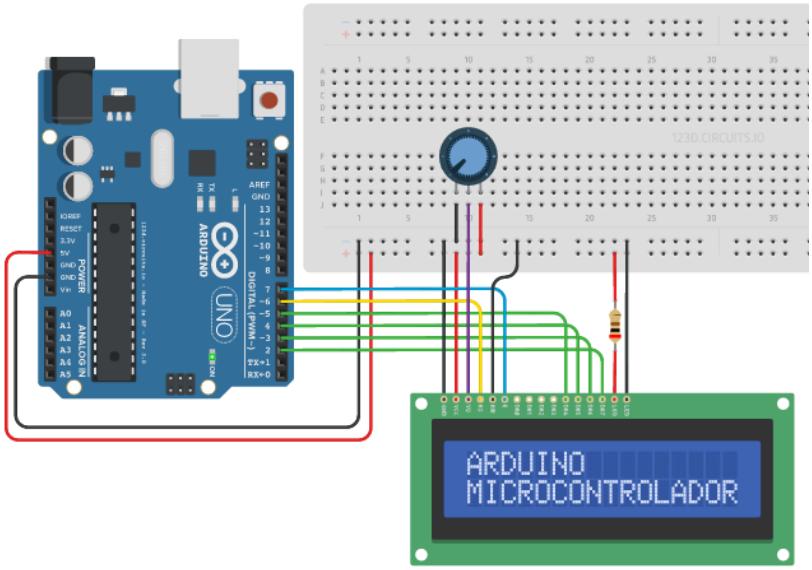


Figura 6 - Circuito da prática 01

```
#define pinoRS 6
#define pinoEnable 7
#define pinoD4 5
#define pinoD5 4
#define pinoD6 3
#define pinoD7 2

#Include <LiquidCrystal.h>

LiquidCrystal lcd(pinoRS, pinoEnable, pinoD4, pinoD5, pinoD6, pinoD7);

void setup() {
   lcd.begin(16,2);
}

void loop() {
   lcd.setCursor(0,0);
   lcd.print("ARDUINO");
   lcd.setCursor(0,1);
   lcd.print("MICROCONTROLADOR");
}
```

Código 2 - Sketch da Prática 01

4.2. Prática 02 – Contador

Nesta prática o display será utilizado para mostrar uma contagem crescente. Para isso, a montagem utilizada será a da Figura 7. O programa descrito no Código 3 deve ser digitado na IDE do Arduino. Neste *sketch* é definida uma variável global chamada *contador*, que será atualizada a cada 1 segundo para realizar a contagem. O valor dessa variável é impresso no *display* utilizando a função $print()$.

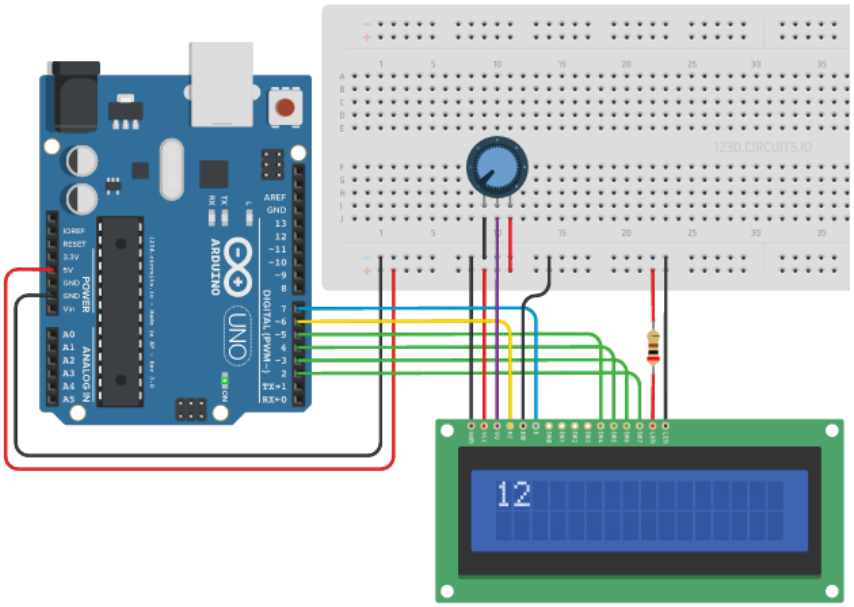


Figura 7 - Circuito da Prática 02

```
#define pinoRS 6
#define pinoEnable 7
#define pinoD4 5
#define pinoD5 4
#define pinoD6 3
#define pinoD7 2

#Include <LiquidCrystal.h>

int contador = 0;

LiquidCrystal lcd(pinoRS, pinoEnable, pinoD4, pinoD5, pinoD6, pinoD7);

void setup( ) {
   lcd.begin(16,2);
}

void loop( ) {
   lcd.setCursor(0,0);
   lcd.print(contador);
   contador++;
   delay(1000);
}
```

Código 3 - Sketch da Prática 02

4.3. Prática 03 – Relógio (Opcional)

O objetivo deste experimento é criar um relógio digital utilizando o LCD. Para realizar a experiência utilize o mesmo circuito da prática 02 e altere o programa para mostrar as horas no formato $hh{:}\,mm{:}\,ss$.

Se quiser aprimorar o código é possível utilizar a interface serial para definir um horário inicial e realizar a contagem após esse ajuste inicial.

5. REFERÊNCIAS

BANZI, Massimo. *Getting Started with Arduino*. 2ª ed. Sebastopol: O'Reilly, 2011.

EVANS, Martin; NOBLE, Joshua; HOCHENBAUM, Jordan. *Arduino em Ação*. 1ª ed. [S.l.]: Novatec, 2013.

HITACHI. *Hitachi HD44780U LCD Datasheet*. Disponível em: <https://www.sparkfun.com/datasheets/LCD/HD44780.pdf>. Acesso em: 8 maio 2016.

MONK, Simon. *Programação com Arduino: começando com Sketches*. 1ª ed. Porto Alegre: Bookman, 2013.